Data Mining with Semantic Features Represented as Vectors of Semantic Clusters

Merwyn Taylor

The MITRE Corporation

mgtaylor@mitre.org

Abstract. Data mining with taxonomies merged with categorical data has been studied in the past but often limited to small taxonomies. Taxonomies are used to aggregate categorical data such that patterns induced from the data can be expressed at higher levels of conceptual generality. Semantic similarity and relatedness measures can be used to aggregate categorical values for cluster based data mining algorithms. Many aggregation techniques rely solely on hierarchical relationships to aggregate categorical values. While computationally attractive, these approaches have conceptual limitations that can lead to spurious data mining results. Alternatively, categorical data can be aggregated using hierarchical relationships and other semantic relationships that are expressed in ontologies and conceptual graphs thus requiring graph based similarity/relatedness measures. Scaling these techniques to large ontologies can be computationally expensive since there is a wider search space for expressing patterns. An alternative representation of semantic data is presented that has attractive computational properties when applied to data mining. Semantic data is represented as vectors of cluster memberships. The representation supports the use of cosine similarity measures to improve the run-time performance of data mining with ontologies. The method is illustrated via examples of K-Means clustering and Association Rule mining.

Keywords: Semantic Similarity, Ontologies, Taxonomies, Semantic Vectors

1 Introduction

Data mining with taxonomies has been studied as an approach to include background knowledge in the mining process. The background knowledge has been used to preprocess data by replacing the original data with more general semantic concepts at arbitrary levels of generalization and augmenting data with all possible more general semantic concepts as defined by a small taxonomy [1]. The benefits of using taxonomies range from smaller search spaces, if the original values are replaced with more general values, to fewer and more intuitive patterns that are expressed at higher conceptual levels. When the original data is replaced with more general concepts, the data is compressed and thus the search space is reduced. When the taxonomies are

adfa, p. 1, 2011.

used to aggregate two or more patterns, fewer patterns are returned and the resulting patterns tend to be more intuitive.

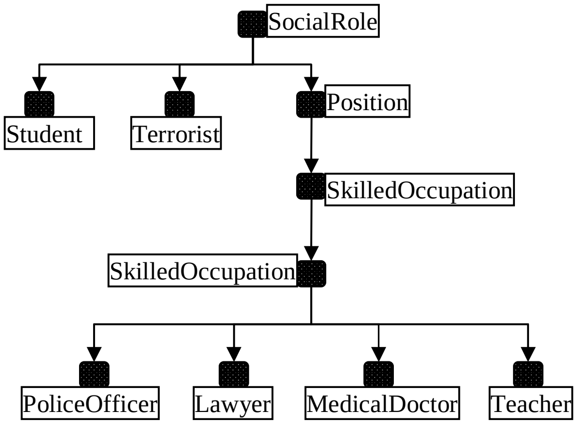

Fig. 1. Portion of the SocialRole branch of the SUMO ontology

Many of the aforementioned techniques are based on shallow taxonomies. Often, the assumption is that if two or more concepts share a common ancestor then the concepts are similar enough to aggregate under the common ancestor. While this assumption may be sufficient for many carefully organized taxonomies, there are occasions in which some portions of ontologies were developed using individual characteristics of concepts to aggregate them and ignoring other characteristics. In such cases, the full semantics of the concepts are not captured in the hierarchical structures. To see this, consider **Fig. 1** which contains a sample of the SocialRole branch of the SUMO ontology. Many of the records used in this study contain values that are subclasses of SocialRole and measuring the semantic distance between these values solely on the basis of the ISA hierarchy is difficult. For instance, Student and Teacher should be grouped together if they appear in database records but in **Fig. 1** Student would be grouped with Terrorist and Teacher would be grouped with Lawyer potentially leading to undesirable results.

Detailed ontologies with relationships between concepts can provide more useful semantics that can be used to more accurately measure semantic similarity. In this study many of the values for a feature are subclasses of a specific class in an ontology. Therefore, fine-grained distinctions that can be computed from non-hierarchical ontological relationships are important. The disadvantage to using the full scope of ontologies is the added search space and computational complexity of aggregating concepts using more semantic information. The problem addressed in this article is that of including hierarchical and non-hierarchical semantic information, expressed in ontologies, in the data mining process while retaining favorable computational properties.

In this article, an approach to using ontologies in K-Means clustering and association rule mining is presented that has favorable computational properties while allowing the full scope of ontologies to be included. The methods involves pre-computing semantic distance and relatedness based on semantic networks, clustering semantic values in which values from individual attributes are clustered separately, and representing the semantic information in attributes as vectors of cluster membership. The computational benefits of this approach are demonstrated in an application of data mining designed to characterize the activities of organizations presented in the World Incident Tracking System (WITS) corpus.

2 Related Work

In [1] Skrikant and Agrawal present their algorithm for generating association rules with taxonomies. In this work, transactions are augmented with information from a taxonomy by identifying all items that are referenced in a taxonomy and traversing the IS-A links to the root of the taxonomy. All concepts encountered along the way are added to the transactions. The association rule induction algorithm is then applied. The result is a collection of association rules that can potentially include concepts from the taxonomy. This approach does not include rich semantic information from ontologies but rather relies on IS-A links to aggregate categorical values. By adding concepts to the transactions, the number of items in a transaction is increased. For large taxonomies, this approach can significantly increase the search space.

In [2] Cheung presents an attribute oriented induction algorithm to generate characterization rules using small rule-based taxonomies defined over categorical attributes. A rule-based taxonomy for attribute A_1 is a taxonomy in which the IS-A links are conditioned on other attributes. The attributed–oriented induction algorithm is a bottom-up mining approach but can apply drill-down operations to specialize rules that are too general. This approach relies solely on taxonomies but uses a more expressive taxonomy. The rule-based taxonomy could contain expressions that more accurately convey the semantics of concepts and thus has the potential to avoid the pitfalls of relying on unconditional IS-A links. The research presented in this article is similar to [2] in that conceptual aggregation is based on the relationships other than unconditional IS-A links. The algorithm described in [2] explores a larger search space than the one discussed in this article.

In [3] Taylor presented a discrimination rule induction approach to using taxonomies for data mining. In this approach, rules are created by repeatedly extending queries using the attributes of a database as constraints and IS-A links to refine the queries. The process is an iterative deepening process in which attributes are initially constrained to the highest level of generalization and repeatedly specialized by traversing down the hierarchies. This approach repeatedly uses a query evaluation optimization technique based on the observation that a query Q’=c+Q derived from query Q can be evaluated by constraining the results of Q to those results that satisfy c without scanning the entire data set. The taxonomy may have to be scanned if c is a hierarchical constraint.

In [4] Zhang present the AVT-DTL algorithm to produce decision trees from partially specified datasets and taxonomies. The algorithm is top-down search in which attributes are constrained to the most general values, in a taxonomy, that are the most informative. Initially all attributes are constrained to the roots of the respective taxonomies defined for the attributes. The algorithm builds decisions trees by repeatedly constraining attributes by replacing classes with their immediate descendants based on how well the set of constraints partitions the classes. This algorithm maintains access to taxonomies and split decisions are based on IS-A links. The algorithm does not use non-hierarchical properties.

In [5] Domingues and Rezenda present an algorithm that applies taxonomies to association rules after the association rules have been created. This algorithm merges association rules that contain antecedents and/or consequences that can be merged if the items share common ancestors in a taxonomy. In [6] Marinica et. al. present a similar post-processing approach to including taxonomies with association rule induction. Both of these approaches rely solely on the IS-A links. The search spaces of [5,6] are not as large since the hierarchical information is not fused with the original records.

In [7,8], Jozefowska presents an approach to discovering frequent patterns from data stored in graph based structures using OWL ontologies with rules. The general approach begins with a user defined context which declares the semantic type of the data to anchor the search. A query is generated using this context. The query is repeatedly refined using the ontology and rules as the basis for extending the query. Queries that are supported by the number of examples that exceed minimum support thresholds are candidates for patterns. The ontology and rules are used to filter queries based on logical consistency checks. The search is pruned by identifying those queries that are logically inconsistent. This approach allows the full semantics of an ontology to be used in the mining process at the expense of repeatedly evaluating the consistency of an expression. Consistency checks can be expensive based on the complexity of the rules and the axioms in an ontology.

The techniques discussed in [1,3-6] rely solely on a taxonomy and are susceptible to the limited semantics expressed in a taxonomy. The research presented in this article differs from [1,3-6] in that additional semantic information is used and the ontology is not repeatedly scanned during the mining process. [2,7,8] demonstrates that using additional semantic information increases the complexity of mining with ontologies. The research presented in this article addresses the complexity issue via a compromise of semantic fidelity. The work presented in [7,8] preserves the full ontology during the mining process, while research presented in this article promotes scanning the ontology prior to data mining to make semantic commitments then subsequently mining the data using the semantic commitments.

3 Modeling Semantic Attributes in Records

Let R = $\{r_1, r_2, \ldots, r_n\}$ represent a set of records containing numerical and categorical attributes. For the remainder of this article, the emphasis will be placed on the categorical attributes. Let Ac = {a_1, a_2, …, a_k} represent the set of all categorical attributes for records in R. Categorical attributes are allowed to have multiple values. Let O = {C, H, P} represent an ontology with a set of concepts C, a set of hierarchical relationships H defined over concepts C, and a set of non-hierarchical relationships P between concepts in C. The set H denotes the set of all IS-A connections between concepts. The set P represents the set of relationships that detail the semantics of concepts beyond that which can be expressed with just IS-A links. Every allowable value for attribute a_i is either a term(s) that has at least one interpretation in ontology O or is a concept(s) in ontology O. The challenge of data mining with ontologies is to find interesting patterns in R expressed as constraints using concepts from C on attributes in Ac.

Conceptual aggregation is the primary goal of data mining with ontologies. Given two or more records in R, the challenge is to determine if there are any semantic relationships between the records and to determine if those relationships are strong enough to aggregate those records to contribute to meaningful clusters or interesting patterns. As mentioned earlier, categorical data can be aggregated by searching for common ancestors and measuring similarity/relatedness based on common ancestors or analyzing the semantics of the concepts via relationships in P and measuring similarity/relatedness base on P. Data mining algorithms tend to iterate over the information in R many times to produce interesting patterns and as such can compute the same results repeatedly. Computing semantic similarity/relatedness for the same set of concepts repeatedly can be computationally expensive.

To address this issue, the semantic similarity/relatedness measures can be cached and similar concepts can be aggregated into clusters based on similarity/relatedness. In doing so, terms and concepts that are similar are pre-aggregated solely on the basis of the semantics of the terms and concepts. The values for attributes in Ac can then be represented as vectors of cluster memberships and records in R can be compared using vector based measures which are computationally more efficient than repeatedly scanning ontologies and repeatedly comparing classes.

Let R_1 = {S#Teacher,…}, R_2 = {S#Student,…}, R_3 = {S#PoliceOfficer,…}, R_4 = {S#SecurityGuard,…} represent records from R in which the semantic attribute a_1 is presented using concepts in the SUMO ontology[1]. Intuitively, records R_1 and R_2 should cluster well together on attribute a_1 and records R_3 and R_4 should cluster well together. Given this observation, the unique concepts for attributes a_i in Ac can be clustered based on semantic similarity/relatedness prior to data mining operations. Let P contain the following statements expressed in F-Logic:

```
S#PoliceOfficer[hasSkill -> CE#Protecting].
S#SecurityGuard[hasSkill -> CE#Protecting].

S#Teacher[hasSkill -> S#EducationalProcess].
S#Student[patient -> S#EducationalProcess].
```

[1] The namespaces have been abbreviated. S is used from SUMO. M is used for MONTY (proprietary). CE is used from CriminalEvent (proprietary).

Given the contents of $R_{1\text{-}4}$ and P above, the following clusters could be created based on semantic connections:

1. S#Teacher, S#Student
2. S#PoliceOfficer, S#SecurityGuard.

The records can then be represented as vectors of cluster membership with R_1={[1,0]…}, R_2={[1,0],…}, R_3={[0,1],…}, R_4={[0,1],…]}. A non-zero value in position *l* denotes that a record has a semantic value that is in cluster *l* for some attribute. Using this alternative representation, records can be compared for similarity using the cosine similarity measure. Let R_5={{S#Teacher,S#PoliceOfficer},…} represent a record with multiple values for attribute a_1. Using the alternative representation, R_5={[1,1],…} represents that R_5 contains values for attribute a_1 that have semantic interpretations that belong to clusters 1 and 2. Comparing R_5 with the other records along the a_1 dimension reduces to cosine similarity calculations instead of multiple semantic distance and semantic relatedness calculations.

To cluster the concepts used in attribute a_i, a hierarchical clustering technique is used. The clustering is based on pre-computed semantic relatedness based on semantic relationships expressed in P and hierarchical based semantic similarity.

Let SR(C_1,C_2) denote the pre-computed semantic relatedness between concepts C_1 and C_2.

$$SR\left(C_{1,}C_2\right) = 1 - \frac{shortestPath(C_1, C_2)}{maxPathLength} \qquad (1)$$

SR(C_1,C_2) is computed using a shortest path traversal from C_1 to C_2 based on the relationships expressed in P. SR(C_1,C_2) is set to a maximum value if the length of the shortest path between C_1 and C_2 is greater than a predetermined maximum distance. This optimization prevents the process from computing relatedness values for concepts that are intuitively not related even if there exists a path between them. Values close to 1 suggest strong relatedness and values close to 0 suggest weak relatedness. SR can be pre-computed since the ontology does not change with records in R. Therefore, when ontology O is used many times for different data mining runs and/or R is large, the cost of pre-computing SR is negligible.

When computing SR(C_1,C_2), the graph traversal should be limited to relationships that convey positive connections between concepts. Since the goal is to aggregate concepts based on similar semantics, those expressions in P that convey semantic differences should be avoided. An example of a relationship to avoid that often appears in semantic networks is “antonym”.

Let SD(C_1,C_2) denote the semantic distance between concepts C_1 and C_2. The semantic distance can be computed using a variety of semantic distance measures that are based on traversing IS-A hierarchies such as those presented in [9,10,11]. The class of techniques mentioned in [9,10,11] is based on finding the Least Common Ancestor (LCS) of a pair of concepts and computing similarity based on the LCS.

The semantic distance measure that is used in this research emphasizes the depth of concepts in a taxonomy. The semantic distance is measured by the path between two concepts (along IS-A links) that includes the LCS. Many semantic distance measures use a uniform distance measure between concepts and ancestors. These measures assume that the semantic distance between child-parent pairs at all depths are uniform. However, the semantic distances between concepts close to the root are greater than the semantic distances of concepts further from the root. This phenomenon has also been observed in [12,13,14]. Given two concepts C_1 and C_2 and their LCS we define the semantic distance between C_1 and C_2 as

$$SD(C_1, C_2) = \sum_{\gamma}^{C_{1-LCS}} \frac{depth(\gamma)}{maxDepth} + \sum_{\mu}^{C_2} \frac{depth(\mu)}{maxDepth} \tag{2}$$

where γ iterates over the concepts on the shortest path from C_1 to the LCS and μ iterates over the concepts on the shortest path from C_2 to the LCS.

Let $SDR(C_1,C_2)$ denote the combination of $SD(C_1,C_2)$ and $SR(C_1,C_2)$.

$$SDR(C_1, C_2) = SD(C_1, C_2)\alpha + (1 - SR(C_1, C_2))\varphi \tag{3}$$

where α and φ denote the contributions of semantic distance and semantic relatedness respectively and both values combine to a value of 1.

The cluster generation algorithm is presented below:

```
For each a in Ac Repeat
  Let U(a) = {unique values of a in R}
  $\theta$ = maximum cluster distance
  Create a cluster for each value in U(a)
  Repeat until all cluster distances > $\theta$
    Merge the 2 closest clusters using the pairwise average of SDR as a measure of cluster distance.
```

The resulting clusters contain classes that are closest to one another based on the contents of T and P and the value of θ. Given the problems of relying on IS-A hierarchies to measure semantic relationships, the semantic quality of the clusters will be largely based on the amount of information available in P. The value of θ can also influence the quality of the clusters but more so based on space and not semantics. Values of θ closer to 1 tend to force classes into clusters simply because they are the closest and not because they share significant semantic overlap. Values of θ closer to 0 tend to produce more coherent clusters since clusters are less likely to be merged together if there is little semantic overlap among the classes. However, if θ is too small, concepts that are naturally similar could appear in different clusters. Determining the most appropriate value of θ is beyond the scope of this research. In this study, θ=0.6.

A hierarchical clustering approach was selected because the clusters produced tend to be consistent between separate executions and thus the mining results will be repeatable. Clustering techniques based on random selections should be avoided for this phase as they tend to produce slightly different clusters between executions.

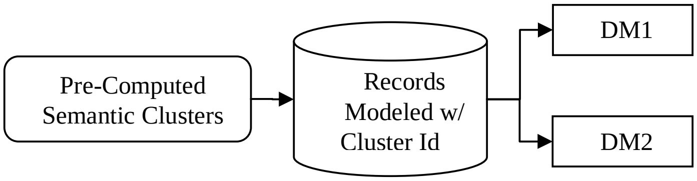

Fig. 2. Proposed diagram for data mining with ontologies.

The contents of P should contain semantic relationships that are sufficient to support reasonable conceptual comparisons along dimensions that are significant for the data mining goals. Those relationships that do not convey information that is relevant to a particular task should be avoided when computing the semantic relatedness values. Furthermore, P should contain expressions that are applicable to the concepts that appear in R. To determine if P is sufficient for a particular data set, one can extract the unique semantic values in R and find relationships in P that are expressed using these values. If many of the semantic values in R are not used in relationships expressed in P, then P may not be sufficient for data mining with ontologies. In this case, P should be augmented with additional relationships that partially detail the semantics of the most frequently used concepts in R.

The relationships in P can represent connections between concepts that vary in significance. For instance, it is not uncommon for semantic networks to contain very generic relationships such as simply “Related-To”. “Related-To” suggests that there exists a relationship between two concepts but the nature of the relationship is not exposed. When constructing the semantic relatedness cache, the type of relationship between concepts can optionally be weighted based on the significance of the relationship expressed.

4 Data Mining Modifications

This section presents applications of the semantic representations presented in the previous section. The general approach to data mining with ontologies proposed in this study is illustrated in **Fig. 2**. The first step is to pre-compute semantic clusters for those attributes that store semantic information. The original records are then modeled using the cluster identifiers to which the original semantic information belongs. Finally, apply the data mining algorithms on the altered records.

4.1 Modified K-Means Clustering

A key component of the K-Means clustering algorithm is the function used to compute the distance between the value for attribute a_i for a record R_j and the value of attribute a_i for the cluster center of cluster G_g. Another component is the function that computes the contribution to the cluster center for attribute a_i. The functions that are used in the K-Means algorithm are presented in this section.

Let CC(G_g, a_i) denote the function that computes the value of the cluster center contributed by attribute a_i. CC(G_g, a_i) returns a vector of length equal to the number of clusters created for attribute a_i. This center can be approximated by the vector average on attribute a_i for all records in cluster G_g.

$$CC(G_g, a_i)[l] = G_g^{a_i}[l] = \frac{\sum_r^{C_c} R_r^{a_i}[l]}{|G_g|} \tag{4}$$

where r iterates over records in cluster G_g and *l* is the *l*th vector position. This definition of cluster center maintains concise semantic centers when every record in a cluster has a non-zero value in the same position. The cluster centers don't drift. When the records have non-zero values in different positions this definition of cluster center maintains separation of semantic information from different clusters. To see this, consider the example presented section 3. In the event that the clustering algorithm places R_1 and R_3 in the same cluster because other attributes are similar enough to support that grouping, the cluster center would be [0.5, 0.5]. In this case, the cluster center is not a concept that connects S#Teacher and S#PoliceOfficer but rather partial contributions of the clusters to which the concepts belong.

In contrast, the semantic center could be computed more precisely by collecting all unique values for attribute a_i and treating the set of values as the center. Calculating distance would then reduce to taking the average distance between the semantic values for a record and those of the cluster center. Another approach would be to determine a semantic center by selecting a concept that is the closest to all concepts in the set of values for attribute a_i for those records in cluster G_g. In both cases, semantic distance/relatedness measures would have to be repeatedly computed.

Let CD(R_j, G_g, a_i) denote the function that computes the distance between a record and a cluster with respect to a single attribute that represents semantic values.

$$CD(R_{j,}G_g, a_i) = 1 - \frac{\sum_l R_j^{a_i}[l] * G_g^{a_i}[l]}{\sqrt{\sum_l R_j^{a_i}[l]^2} + \sqrt{\sum_l G_g^{a_i}[l]^2}} \tag{5}$$

where X[*l*] is the value of *l*th vector element representing cluster membership for semantic attribute a_i.

With these two functions, the K-Means algorithm can be implemented to efficiently include information from an ontology without repeatedly scanning an ontology to compute the distance between two more concepts.

4.2 Modified Association Rule Mining

Association rule induction was first introduced by Agrawal in [15]. Given records of transaction, the goal of association rule induction is to produce rules of the form X -> Y where X is a combination of items in the records and Y is one or more items in the records. In [2], Agrawal, introduced taxonomies to association rule induction. The approach presented involves merging hierarchical information with the original data from the records and execute the original association rule induction process. In this study, instead of merging semantic information with the original data, the original data is replaced with cluster membership indices. The attribute value pairs are converted to items by concatenating the attribute identifiers with the cluster identifiers.

Consider the example records from section 3. R_1 would be modeled as {a_1_1,…}. This set denotes that record R_1 contains an item in which the value for attribute a_1 is a term or concept that is assigned to cluster 1 from the set of clusters created for the unique semantic values extracted for attribute a_1. The item "a_1_1" encompasses the semantic information for a single cluster of semantically related values. Item "a_1_1" encompasses the hierarchical similarity that is often used in other approaches and it encompasses other relationships that relate concepts to one another. As a result, finding records with semantic values similar to those in R_1 is reduced to finding records with the item a_1_1. Following this modeling approach, R_2 would be modeled as {a_1_1,…}, R_3 as {a_1_2,…}, R_4 as {a_1_2,…}, and R_5 as {a_1_1, a_1_2, …}. The typical association rule mining algorithms are then applied to these records.

There are two main benefits of this approach to association rule mining with ontologies. First, there are fewer items per record compared to the approach of adding semantic data to the original items as in [2]. Fewer distinct items reduce the search space for association rule induction and thus leads to reduced execution times. The semantic information encoded in the items includes hierarchical relationships as well as non-hierarchical relationships that often convey richer semantics of the terms and concepts that appear in the records. As a result, association rules can be induced using more information from the ontologies while not incurring the computational cost of repeatedly calculating semantic similarity between sets of concepts.

5 Performance Results

To demonstrate the potential of this approach, two data mining algorithms were implemented to find patterns of activity for organizations mentioned in reports from the WITS corpus. The documents describing violent criminal events attributed to a single organization were parsed and modeled as vectors of varying numerical and categorical information. The categorical information includes instrument type, victim type, target type, location, and event type. The SUMO ontology was used as the basis of a proprietary ontology that was used in this study. The base ontology was augmented with relationships that partially characterize the semantics of many of the concepts mentioned in the WITS corpus. The bulk of the concepts mentioned in the WITS corpus with respect to the targets and victims of criminal activity are social roles. The SocialRole branch of the SUMO ontology is too shallow to rely on typical semantic distance measures alone. Therefore, the proprietary ontology was updated to include semantic relationships on many of the social roles expressed in the WITS corpus.

Eight organizations were studied in this research. Some of the organizations were very active while others were moderately active. The number of records found for each organization reflects the number of events reported in the WITS corpus that were attributed to the organizations.

For this experiment, the values of α and φ used in Equation 3 were both set 0.5. As a result, the values $SD(C_1,C_2)$ & $SR(C_1,C_2)$ equally contribute to the semantic comparisons of concepts that appear in the records. The decision was made to value SD & SR equally because P was not fully developed at the time of this publication. Note that the value for $SR(C_1,C_2)$ is 0 if there is not a path from C_1 to C_2 in P. Therefore, the value $SD(C_1,C_2)$ is the major contributor to Equation 3 when C_1 and C_2 are not sufficiently related.

Records	w/o Semantic Clusters	w/ Semantic Clusters
21	40	15
55	76	26
106	98	31
112	90	32
179	97	35
368	164	52
1225	136	40
1573	194	57

Table 1. Number of distinct semantic items per data set availabe to the data mining algorithms

Table 1 lists the number of distinct values for the original representation of documents from the WITS corpus and vector based representation. Using the semantic clusters to represent the semantic values reduces the number of unique items that are available to the data mining algorithms.

Table 2 includes the execution times in milliseconds for two implementations of the K-Means clustering algorithm using ontologies applied to the records produced from eight sets of documents from the WITS corpus. The 1st column lists the number of records extracted per group. The K-Onto column lists the execution times for K-Means without the semantic clusters representation and the K-Vects column lists the execution time for K-Means using the semantic clusters representation. The K-Vects times include the time to produce the semantic clusters and the time to execute the K-Means algorithm. The Speed-Up column lists the performance improvements achieved using the semantic clusters. For all record sizes, there is a clear performance boost. The highest performance gain is realized with the largest dataset. The performance boost does not uniformly increase with record size in part because the number of interesting patterns in records is not always a function of size but rather a function of distributions of values.

Records	K-Onto (ms)	K-Vects (ms)	Speed-Up
21	196	25	7.84
55	949	119	7.97
106	2922	281	10.39
112	2328	282	8.25
179	5985	506	11.82
368	20648	2348	8.79
1225	165563	20820	7.95
1573	607270	20302	29.91

Table 2. Execution times in milliseconds for K-Means using alternate approaches to include ontologies to measure semantic similarity.

Records	Attribute Cluster Creation (ms)	K-Vects (ms)	% Cluster Creation
21	17	25	68%
55	74	119	62%
106	163	281	58%
112	143	282	51%
179	161	506	32%
368	952	2348	41%
1225	907	20820	4%
1573	1474	20302	7%

Table 3. Attribute semantic cluster creation times compared to total record clustering times.

In **Table 3** the attribute cluster creation times are compared to the total records clustering times. For small data sets, the attribute cluster creation times dominate the total processing times. For larger data sets, the attribute cluster creation times account for smaller percentages of the total record clustering times. The values in **Table 3** suggest that a significant amount of time is consumed performing the initial semantic analysis of the terms and or concepts that are allowable values for the semantic attributes. While the initial semantic analysis can dominate the overall executions, the values in **Table 2** show that this initial cost is more than offset by the reduction in clustering with ontologies afterwards.

A manual inspection of the cluster assignments produced by K-Onto and K-Vect when applied to the data set containing 21 records revealed that many of the documents were clustered together by both approaches. An examination of the cluster centers revealed similar values.

In **Table 4** the execution times of the two Association Rule induction algorithms are compared. AR-Onto is an association rule induction algorithm in which attribute value pairs are treated as items. The ontology is accessed to find concepts that are related to semantic values in the records, and the related concepts are added to the records as new items. For each semantic value, the ontology was accessed to find ancestors of concepts that were within 3 hops away following the IS-A links and those concepts that have a pre-computed relatedness value less than 0.26. AR-Vects is an association rule induction algorithm in which attribute value pairs are treated as items, however, semantic attribute values pairs are represented as cluster identifiers as described in section 4.2. The times reported in the AR-Vects column include the time of pre-clustering plus association rule induction and the association rule induction times separately in parentheses. The results indicate that the benefits of pre-clustering semantic information are only realized in the 2 smallest datasets and the largest dataset where the speed-ups are significantly greater than 1. There is no run-time benefit to pre-clustering for the other data sets because the cost of pre-clustering is too large.

Records	AR-Onto Time (ms)	AR-Vects Time (ms)	Speed-Up
21	79	25 (8)	3.16 (9.8)
55	163	96 (22)	1.7 (7.4)
106	170	200 (37)	0.85 (4.6)
112	226	208 (65)	1.1 (3.5)
179	203	228 (67)	0.9 (3.0)
368	325	1082 (130)	0.3 (2.5)
1225	3112	2700 (1793)	1.1 (1.7)
1573	1177908	2220 (746)	530.6 (1579)

Table 4. Execution times for Association Rule Induction with ontologies. Times in parentheses are times – preclustering execution times

The results of pre-clustering the semantic values are available to both data mining techniques. If we compare the execution times of AR-Onto and AR-Vect, minus pre-clustering times, the AR-Vect algorithm runs faster than the AR-Onto algorithm. The difference in performance is attributed to the reduced number of distinct items available to AR-Vect as listed in **Table 1** and the total number of item sets that are tested as listed in **Table 5**. The significant difference in performance on the dataset with 1573 records is attributed to 269393 item sets that were tested compared to testing only 489 item sets. Many of the association rule induction optimization techniques could be applied to AR-Onto to reduce the run-times.

AR-Vect finds association rules involving closely related concepts that are allowable values for the same attribute. AR-Onto finds associations involving individual concepts but can find associations involving related concepts if all permutations of related concepts are included as items which requires more search. A manual analysis of the patterns produced by AR-Onto and AR-Vect applied to the data set with 21 records revealed that the algorithms produced some patterns that were semantically equivalent.

Records	AR-Onto Item Sets	AR-Vects Item Sets
21	1225	220
55	1824	361
106	1138	383
112	1307	512
179	888	325
368	601	300
1225	1240	929
1573	269393	489

Table 5. Total number of item sets generated and tested

AR-Vect can efficiently find item sets that AR-Onto cannot find efficiently. If Val_1 and Val_2 are sufficiently related, but the 1-item item set $\{\text{Val}_1\}$ does not satisfy minimum support requirement, the 2-item item set $\{\text{Val}_1,\text{Val}_2\}$ will never be explored using the optimized association rule algorithm in which all 1-item item sets have minimum support and the 1-item item sets are used to extend multi-item item sets. To find item sets containing both Val_1 and Val_2 all 1-item item sets will have to be considered, although only those combinations of items that are closely related should be combined in multi-item item sets. This expands the search space which increases execution times.

In this study, multiple data mining algorithms are used to characterize the activities of organizations. Since two data mining algorithms are used in this study, and pre-clustering significantly improves the run-time performance of K-Vect the cost of pre-clustering can be ignored in **Table 4** when comparing the run-time performances of the AR-Onto and AR-Vect. If multiple data mining algorithms are going to be used with ontologies, then performing a semantic analysis of those dataset attributes that are semantic in nature prior to executing the data mining algorithms can significantly improve the run-times of the data mining processes collectively.

The records that are produced from the WITS corpus can contain multiple values for some of the semantic attributes. An interesting property of using the semantic vectors for clustering, as in K-Vect, is that the cost of comparing records with multiple values for attributes with semantic values does not change. In contrast, methods that maintain the original values will have to compare all values for a given feature to determine similarity. As the number of values per feature increases, the cost of computing similarity increases.

6 Conclusion

The approach to data mining with ontologies presented in this article has favorable computational properties but presents a semantic challenge. The semantic clusters that are created prior to executing the data mining routines encapsulate semantic information via cluster membership. However, since these clusters can represent two or more semantic classes and the cluster identifiers are used during the mining processes, any relationships that may exist between individual values from different attributes may not be distinguishable. For example, cluster 1 in section 3 represents S#Teacher and S#Student for attribute a_1. If an interesting relationship between S#Teacher and values from other attributes exist in R, it may be difficult to isolate the relationship since S#Teacher is always coupled with S#Student. This potential loss of semantic fidelity is a trade-off that accompanies the computational advantages of the methods present in this article.

The cosine distance measure as implemented in the K-Means example assumes maximum distance for concepts in different clusters. To see this consider records R_1 and R_3 in section 3. The cosine based distance measure applied to the vectors for attribute a_1 in these records would return a value of 1 suggesting that S#Teacher has no relationship to S#PoliceOfficer. In fact, there is a weak relationship between these two concepts in the SUMO ontology but given the semantic commitments that are made, during the attribute pre-clustering phase, the relationship is ignored. If the relationships between concepts in different clusters are important to a particular application, the cosine based distance function could be altered to include average semantic distances between clusters when one vector has a non-zero value in position *l* and another vector has a zero value in the same position. The vector for attribute a_1 in R_1 could be replaced with $[1, \in]$ and that of R_3 could be replaced with $[\in, 1]$ where $\in$ is the computed cluster distance between clusters 1 and 2 of attribute a_1. With this modification, the cosine based distance between R_1 and R_3 along the a_1 dimension would be < 1 if $\in > 0$. This modification would require pre-computing cluster distances after the attribute clusters have been created.

The types of the relationships that justify clustering concepts together are not exposed in the vector of cluster membership model. As such, this approach does not lend itself well in environments where the particular relationships between concepts are more important than the fact that concepts are strongly related.

References

1. R. Srikant, R. Agrawal. Mining Generalized Association Rules. In Proc. 21st Intl. Conf. on Very Large Databases. 1995
2. D. Cheung, H. Hwang, A Fu., and J. Han. Efficient Rule-Based Attribute-Oriented Induction for Data Mining. Journal of Intelligent Information Systems. Vol. 15, No. 2, pp 175-200, 2000.

3. M. Taylor, K. Stoffel, and J. Hendler. Ontology-based Induction of High Level Classification Rules. In Proc SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, 1997.
4. J. Zhang, and V. Honavar. Learning Decision Tree Classifiers from Attribute Value Taxonomies and Partially Specified Data. In Proc. 20th Intl. Conf. on Machine Leanring (ICML-03). Washington DC.
5. A. Domingues, and S. O. Rezende. Using Taxonomies to Facilitate the Analysis of the Association Rules. In Proc. of ECML/PKDD 2005 - The 2nd Intl. Workshop on Knowledge Discovery and Ontologies (KDO 2005)
6. C. Mirinica, E. Guillet, and H. Briand. Post-Processing of Discovered Association Rules using Ontologies. In Proc. of the 2008 IEEE Intl Conf on Data Mining Workshops (ICDMW-08). Washington DC
7. J. Jozefowska. The Role of Semantics in Mining Frequent Patterns from Knowledge Bases in Description Logics with Rules
8. J. Jozefowska. Frequent Pattern Discovery from OWL DLP Knowledge Bases
9. P. Resnik. Using Information Content to Evaluate Semantic Similarity. In Proc. 14th Intl. Joint Conf. Artificial Intelligence (IJCAI-95), Montreal, Canada. 1995
10. C. Leacock and M. Chodorow. Combining Local Context and WordNet Similarity for Word Sense Identification. In C. Fellbaum editor, WordNet: An Electronic Lexical Database. The MIT Press, Cambridge, MA, chapter 11, pages 265-283
11. D. Linn. Using Syntactic Dependency as Local Context to Resolve Word Sense Ambiguity. In Proc. 8th Conf. of the European Chapter of the Association for Computational Linguistics (ACL, EACL-1997), pages 64-71, Madrid, Spain.
12. J. Ge and Y. Qiu. Concept Similarity Matching Based on Semantic Distance. In SKG 2008 Proc. 4th Intl Conf. on Semantics, Knowledge, and Grid, 2008.
13. V. Cordi, P. Lombardi, M. Paolo, M. Martelli, and V. Mascardi. An Ontology-Based Similarity between Sets of Concepts. In WOA 2005 Proc of Workshop From Objects to Agents, 2005.
14. H. Nguyen. New Semantic Similarity Techniques of Concepts Applied in the Biomedical Domain and WordNet. Masters Thesis, The University of Houston Clear Lake, 2006.
15. R. Agrawal, T. Imielinski, and A. Swami. Mining Association Rules between Sets of Items in Large Databases. In Proc ACM SIGMOD Intl. Conf. Management of Data, Washington DC, 1993